Stochastic routing in large grid-shaped quantum networks

Cuong Le Quoc
ENST
qle@enst.fr

Patrick Bellot
ENST
bellot@enst.fr

Akim Demaille
EPITA Research and
Development Laboratory (LRDE)
Akim.Demaille@lrde.epita.fr

***Abstract*— This paper investigates the problem of secret key transmissions for an arbitrary Alice-Bob pair in Quantum Key Distribution (QKD)-based networks. We develop a realistic QKD-based network framework and we show that the key transmission problem on such a framework can be considered as a variant of the classical percolation problem. We also present an adaptive stochastic routing algorithm to protect users from inevitable eavesdroppers. Simulations were carried out not only to validate our approach, but also to compute critical parameters to ensure security. These results show that large quantum networks with eavesdroppers do provide security.**

Most applications transport secret keys over the Internet using Public Key Infrastructure (PKI). PKI relies on assumptions about the calculation power of eavesdroppers and the non-existence of effective algorithms for certain mathematical "hard" problems. This is not enough for some applications. Quantum Key Distribution (QKD) technology provides proved unconditional security in the key transmissions [1], [2]. It has however several limitations, most prominently throughput and range [3], [4]. This makes more complex the construction of large QKD-based networks enabling perfectly secret key exchange between members. Besides, large networks are vulnerable, and some intermediate nodes may be controlled by eavesdroppers.

This paper studies a model of partially compromised QKD network in which two members want to establish a common key with the near certainty that this key will not be eavesdropped. The contributions are (i) a model of partially compromised QKD networks, (ii) a proposal based on stochastic routing to augment the secrecy, (iii) the use of percolation theory techniques to demonstrates how near-certainty can be achieved.

The remainder is organized as follows. In Section I, we introduce the context and define our problem statement. In Section II, we present a first algorithm and analyze it using percolation theory. In Section III, we present a new secret key agreement scheme that could be considered as an extension of our first proposal. Relation with other works is presented in Section IV and we conclude in Section V.

I. A PROPOSED QUANTUM NETWORK FRAMEWORK

We first state the context of our work (cryptography, QKD links and networks, stochastic routing), then formulate our proposal, and introduce percolation theory as the tool we use to evaluate some critical parameters.

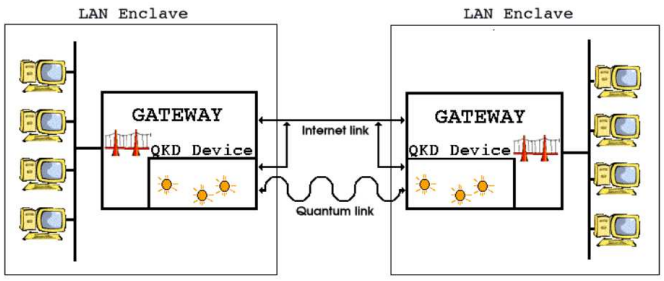


Fig. 1. A single QKD Link

A. Cryptography basics

Suppose Alice wants to transmit a secret message M to Bob over some public network, say the Internet. According to Claude Shannon's information theory [5], a cipher is perfectly secret if the cipher-text C reveals no information about the message M, i.e., if $I(C, M) = 0$. The one-time pad cipher can help Alice and Bob to share perfect secrecy provided that they have a common secret key K that is as long as the message:

- Alice sends K to Bob, K has the same size as M.
- Alice uses K as an one-time pad key to cipher the private message: $C = M \oplus K$ and sends it.
- Bob deciphers C: $M = C \oplus K$.

The question is: how can Alice send K to Bob with absolute secrecy?

B. QKD links

Quantum mechanics prove that a QKD link is absolutely secure. It is already implemented in some realistic applications [1], [2], [3], [4]. Such a link consists of two elementary links (see Figure 1):

1) A classical link (Internet, phone, etc.) to exchange control information for treating quantum data according to a chosen QKD algorithm.
2) A direct quantum link to transport quantum particles, e.g., photons. Quantum data is extracted from these particles and processed to gain inviolable information to build the secret key.

QKD links are a perfect means for Alice to send the key K to Bob. Unfortunately, QKD links provide limited transmission rate and short distance. The fastest rate of single QKD links is 1 Mbps over a 750 m free-space link. As for distance, the

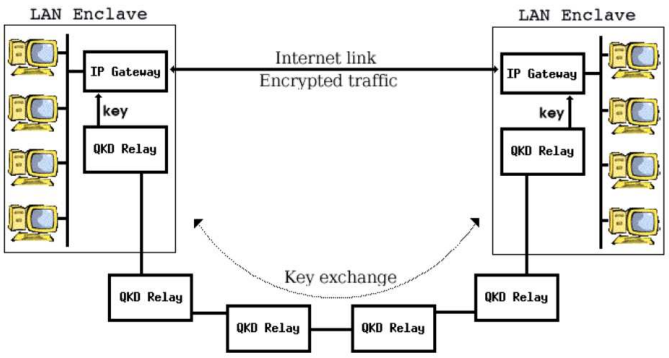


Fig. 2. A long QKD link using relays

current record is 150 km in fiber and 23 km in free space, with a typical rate of 1 Kbps or so.

To overcome the distance limitations, one can use a chain of QKD data relays (see Figure 2). A QKD data relay is not a quantum repeater, it is a terminal that establishes a QKD link with the previous element of the chain and another with the following elements:

- Relay k establishes an encrypted communication, a QKD link, with relay $k-1$;
- Relay k receives encrypted data from relay $k-1$;
- Data is decrypted and stored in the memory of relay k;
- Relay k establishes an encrypted communication, a QKD link, with relay $k+1$;
- Data in memory is encoded and sent to relay $k+1$.

To improve the transmission rate, one needs several QKD routes. In short, building a quantum network on top of QKD links solves throughput and range limitations, but in turn compromises the security by making it more likely that some nodes are eavesdropped.

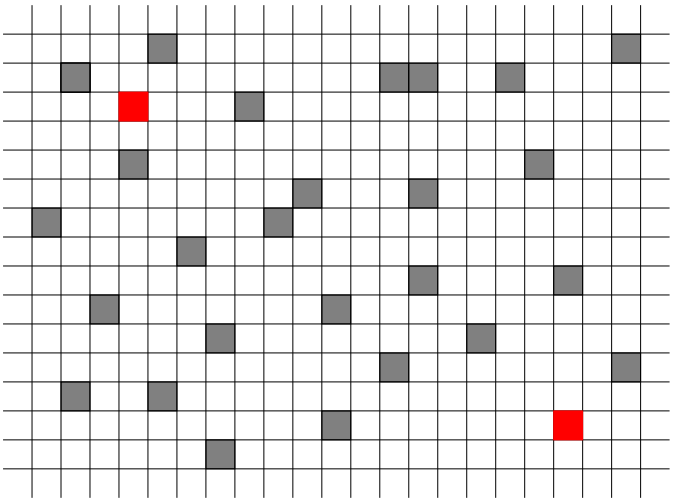
How can we counter the likeliness that some nodes of the network are compromised? How can we be resilient to message interception?

C. Stochastic routing

Traditional routing algorithms, such as those used on the Internet, are mostly deterministic. Because they are tailored to be efficient, one can guess the route that will be taken with a high probability even though there are an almost infinite number of routes connecting two points of the Internet. This compromises security.

Stochastic routing addresses this problem [6], [7]. Its basics are simple: every packet is routed independently, and each time a node has to send (or forward) a packet, it randomly chooses one of its neighbors, not necessarily the most "efficient" one. Of course the selection is slightly biased so that eventually the message is delivered to its destination.

Counter-interception measures are then simple to design: Alice splits key K into n parts K_i so that the whole set is needed to compute K again. One well known method consists in computing K_i such that $K = \bigoplus_{i=1}^{n} K_i$ (bit-wise or). Even if Eve intercepts all the messages but one, the key K is safe.



Nodes are represented by squares (not crossings). White nodes are safe, and gray nodes are attacked and controlled by Eve. Alice and Bob are two random safe nodes.

Fig. 3. Two-dimensional lattice network

Obviously stochastic routing will not work properly if the physical network features pivotal nodes.

D. Quantum network topology

The Internet topology features some backbone nodes where Eve can predict with high probability that packets will pass, provided she knows the source and destination. Such a topology does not suit quantum networks: to ensure that there are many routes, each node should have several (more than two) neighbors.

The poor distance covered by today's QKD-links also influences the design of topology. Although the maximum length of quantum links is about 150 km, because of rates and costs, 30 km-long quantum links are more likely to be used.

As a first step in studying QKD networks and appropriate routing algorithms, we will approximate their topology as a simple square grid, a 4-connected lattice (Figure 3). The grid will be large enough so that we can neglect nodes on the boundaries (Alice and Bob will be far from them). This roughly models a large region meshed with QKD links.

We can now state our problem.

E. Problem statement

Alice wants to send a message to Bob securely. We propose to deploy stochastic algorithms to route split messages on quantum networks in order to augment the resilience to eavesdropping.

The network is modeled as a 2-dimensional 4-connected grid. Eve can control of any node (except Alice and Bob) with probability $p_a \in [0,1]$. These nodes are called *attacked* or *unsafe* nodes; the others are *safe*. Alice and Bob do not know whether a node is safe, but they know the probability $p_s = 1 - p_a$ that a node is safe. Alice splits the message into

n parts sent to Bob using stochastic routing: each message is transported via a random path. Eve needs every single part to decipher the message. The key question is:

How many messages must Alice send so that at least one message is not intercepted? Percolation theory helps addressing that question.

F. Percolation theory

Percolation theory, introduced in 1957 by John M. Hammersley and Simon R. Broadbent, originally aims at modeling the propagation of a fluid through a medium with random permeability. It is one of the simplest models exhibiting a phase transition: there is a critical point at which the global system properties brutally change. Percolation theory provides a nice theoretical framework to study complex systems which present critical thresholds in their state evolution [8], [9].

We focus on 2-dimensional percolation theory on a regular graph $G = \mathbb{Z}^2$, with vertex set V and edge set E. Let the vertices be independently *open* with probability $p \in [0, 1]$ or *closed* with probability $1 - p$. Consider a path π in G as a sequence $\pi = v_1, v_2, \ldots$ of vertices such that for all $i \geq 1$, v_i and v_{i+1} are adjacent. A path *open* iff all the vertices v_i are open. Since we consider open *vertices* (as opposed to edges), this model is called *site percolation*.

The set of open vertices forms a random subgraph of G. The original question of interest is whether the connected cluster of the origin in that subgraph is finite or infinite. This is equivalent to the existence of an unbounded open path starting from the origin.

We denote by $u \leftrightarrow v$ the existence of an open path between two vertices $u, v \in V$. The *open cluster* $C(v)$ of the vertex v is the set of all vertices which are connected to v by an open path:

$$C(v) = \{u \in V : u \leftrightarrow v\}$$

The central quantity of the percolation theory is the *percolation probability*, the probability that the cardinality of $C(v)$ is infinite. It depends on p, the probability that a vertex is open:

$$\theta(p) = \{\Pr(|C(v)| = \infty)\}$$

The most interesting property of percolation models is that $\theta(p)$ exhibits a threshold value $p_c \in [0, 1]$ and a brutal phase transition: the existence of an infinite connected cluster on the system appears (see Figure 4). By definition, when $p < p_c$ all the open clusters are finite. But for $p > p_c$ there is a positive probability that an infinite cluster exists. Formally, the *critical probability* is defined by:

$$p_c = \max\{p : \theta(p) = 0\}$$

Even if the percolation theory is based on statistical models which were initially used to describe critical phenomena in physics, its polyvalence and efficiency in characterizing non-linear phenomena led the scientific community to use this theory to model complex systems such as biological systems, social networks and economic systems.

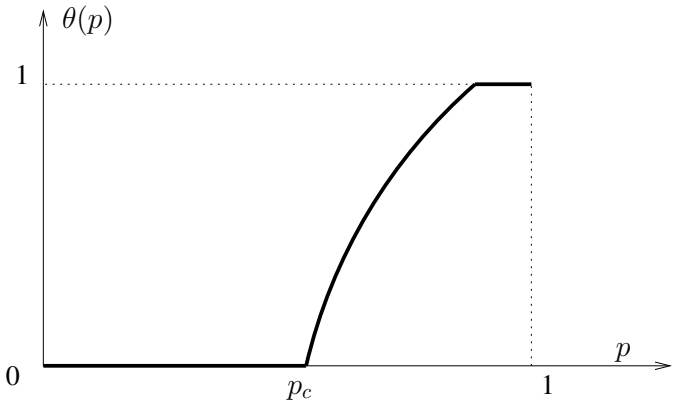


Fig. 4. Correlation between open probability p and percolation probability $\theta(p)$.

II. A simple routing algorithm

A. Percolation-based modeling

There are several similarities between site percolation (Section I-F) and our problem. Both take place on a graph and deal with problems arising from the uncertainty about the quality of nodes. Open nodes and paths in site percolation correspond to safe nodes and paths in our context. But, there is a subtle difference: closed nodes *do not* allow the fluid to pass, and they are identified as such, whereas an *unsafe* node still allows messages to pass through. There is no way to identify whether this node is safe.

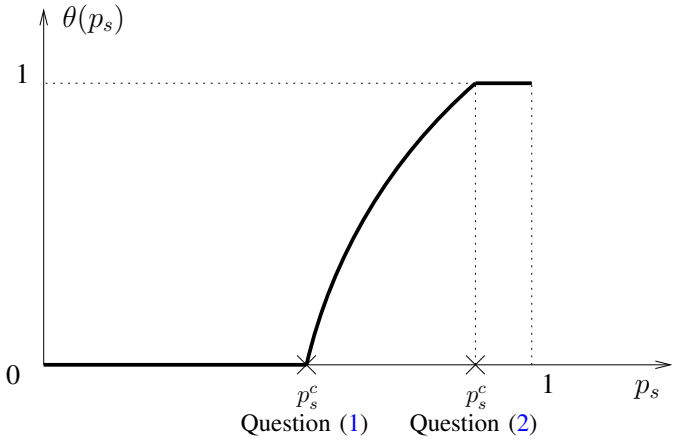
The focus is also different. Percolation theory was built to study the existence of a giant connected cluster in the graph after faulty vertices (or edges) have been randomly chosen and removed. If both Alice and Bob belong to it, then there exists at least one open path between them. At first glance, one may think using the site percolation results in answering the following question:

> *How big can the probability p_s be before the existence of a safe path from two arbitrary safe nodes in the network is lost?* (1)

Though question (1) is similar to the original question of site percolation, it is not the same. In the percolation problem, the value of critical probability permits the existence of a few isolated open nodes that do not belong to the giant open connected cluster. In our context, if Alice or Bob is not connected communication is impossible. Therefore we cannot immediately reuse the traditional results of site percolation. Nevertheless, its methods are relevant to our problem because they are able to deal with the randomness of faulty nodes, and they can help to find our own critical values.

In fact, Question (1) is irrelevant. The question that can help us go to our final goal is:

> *How big can the probability p_s be before the existence of a* safe *path from two arbitrary safe nodes in the network is no longer almost certain?* (2)



In the site percolation on $\mathbb{Z}^2$, the critical value of the open probability p_c is about 0.6. We are interested in the critical probability p_s^c such as any two sites are almost certainly connected. p_s^c is about 0.91 on $\mathbb{Z}^2$ (Figure 6 and Figure 7).

Fig. 5. The critical probability of Question (1) and Question (2).

If we denote by $\theta(p_s)$ the probability that a safe path exists between two safe nodes in the network, then the difference between (1) and (2) is shown in Figure 5.

The answer to Question (2) is the *critical probability* p_s^c that can be defined as follows:

$$\theta(p_s) = \begin{cases} 1, & \text{if } p_s \geq p_s^c \\ \tau : 0 \leq \tau < 1, & \text{if } p_s < p_s^c \end{cases}$$

or:

$$p_s^c = min\{p_s : \theta(p_s) = 1\}$$

Obviously, Question (2) is a variant of the percolation problem. We can use the same statistical models to compute p_s^c: about 0.91 on $\mathbb{Z}^2$ (Figure 4). However, the existence of safe paths does not guarantee they can be found efficiently. Alice and Bob have no way to be certain whether a path is safe, i.e., there is no routing algorithm that finds a safe path. In fact, it is obvious that if $p_s > p_s^c$, then a brute-force routing algorithm can ensure that among its returned paths, there is at least one safe path. However, between two nodes in our network, there are infinitely many routes: brute-force routing is impossible. We need an algorithm that computes a finite number of paths so that at least one of them is safe. And it is desirable to be resilient to variations of network state due to the randomness of attacked nodes chosen by Eve.

We use the following notations:

- p_s: (*safe probability*) probability that a node is safe.
- p_a: (*unsafe* or *attacked probability*) probability that an arbitrary node is controlled by Eve. $p_a + p_s = 1$.
- π_k the route (path) taken by message k.
- $\theta(p_s)$: given p_s, the probability that at least one message is safe.
- p_s^c: (critical value of p_s) $p_s^c = \min\{p_s : \theta(p_s) = 1\}$.
- $\alpha(\pi_1, \ldots, \pi_k)$: probability that at least one of these paths is safe.
- $\beta(\pi_1, \ldots, \pi_k)$: probability that all these paths are safe.
- $\mid \pi \mid$: number of *intermediate* nodes in π.
- $\mid \pi_1, \ldots, \pi_k \mid$: number of common nodes between $\pi_1, \ldots, \pi_k$.

Thus, with a given path π containing $\mid \pi \mid$ intermediate nodes, we can compute the probability that this path is safe:

$$\alpha(\pi) = \beta(\pi) = (1 - p_a)^{|\pi|} = p_s^{|\pi|}$$

We consider the value of $\theta(p_s) = \alpha(\pi_1, \pi_2, \ldots \pi_n)$:

$$= \sum_{1 \leq i \leq n} \alpha(\pi_i) - \sum_{\substack{1 \leq i,j \leq n \\ i \neq j}} \beta(\pi_i, \pi_j) + \sum_{\substack{1 \leq i,j,k \leq n \\ i \neq j, j \neq k, k \neq i}} \beta(\pi_i, \pi_j, \pi_k) - \cdots$$

$$= \sum_{1 \leq i \leq n} (1 - p_a)^{|\pi_i|} - \sum_{\substack{1 \leq i,j \leq n \\ i \neq j}} (1 - p_a)^{|\pi_i| + |\pi_j| - |\pi_i, \pi_j|} + \cdots$$

$$= \sum_{1 \leq i \leq n} (p_s)^{|\pi_i|} - \sum_{\substack{1 \leq i,j \leq n \\ i \neq j}} (p_s)^{|\pi_i| + |\pi_j| - |\pi_i, \pi_j|} + \cdots$$

This is too difficult to evaluate explicitly. So, we use empirical and statistical methods from percolation theory to evaluate $\theta(p_s)$ for the different values of p_s and n. To this end, we need to compute the paths π_i, i.e., we need a genuine stochastic routing algorithm that can return n paths for n messages. We will only consider the cases that have $p_s > p_s^c = 0.91$.

By definition, we have:

$$\lim_{n \to \infty} \alpha(\pi_1, \ldots, \pi_n) = \begin{cases} 1, & \text{if } p_s > p_s^c \\ \tau : 0 \leq \tau < 1, & \text{if } p_s < p_s^c \end{cases}$$

As already mentioned, a brute-force routing algorithm provides infinitely many paths, one of which is almost surely safe. However, this algorithm cannot be implemented. In fact, we look for a routing algorithm and a value $N_0 < \infty$ as small as possible such that:

$$\forall n \geq N_0 : \alpha(\pi_1, \ldots, \pi_n) = 1$$

or more precisely:

$$\alpha(\pi_1, \ldots, \pi_n) = \begin{cases} 1, & \text{if } n \geq N_0 \\ \tau : 0 \leq \tau < 1, & \text{if } n < N_0 \end{cases}$$

B. An adaptive stochastic routing algorithm

We propose an adaptive stochastic routing algorithm and then seek its value N_0 using empirical methods.

The network is a two dimensional mesh $n \times n$, in which each node could be independently eavesdropped with probability $p_a = 1 - p_s$. We limit our attention to the cases of $p_s > 0.91$ so that the connectivity of all the safe nodes is guaranteed. Each node is identified by its coordinates $(i, j) : i = 0, \ldots, n - 1 \ ; \ j = 0, \ldots, n - 1$. The distance between two nodes is defined by:

$$d[(i_1, j_1); (i_2, j_2)] = |i_2 - i_1| + |j_2 - j_1|$$

Resilience is important for an algorithm facing many situation variations caused by the random choices of eavesdroppers. Here, we informally describe an adaptive stochastic routing. When Alice wants to send a message to Bob, she computes the next-hop probability for each neighbor to be the next

step. These next-hop probabilities are determined according to the correlation of coordinates between each neighbor and Bob. To ensure that the message can finally reach Bob, we give a higher probability to nodes closer to Bob. Then Alice randomly chooses one of her neighbors to forward the message, according to these probabilities. Thus, some nodes are more likely to be selected, but nothing is sure. Anyone that subsequently receives the message would do the same thing and the chain of communication would continue to Bob.

Candidate selection and probability assignment can vary. We propose a simple way:

- All the neighbors of the current node are candidates.
- Assume that there are m candidates on the list.
  1) Sort the candidates by decreasing distance to Bob. After sorted, let d_i, be the distance from candidate i to Bob. We have:
  $$\forall i = 1, \ldots, m-1 : d_i \geq d_{i+1}$$
  2) Compute $w_1, \ldots, w_m$:
  $$w_i = \begin{cases} 1, & \text{if } i = 1 \\ w_{i-1}, & \text{if } i > 1 \wedge d_i = d_{i-1} \\ w_{i-1} + 1, & \text{if } i > 1 \wedge d_i > d_{i-1} \end{cases}$$
  3) Compute the next-hop probabilities $\Pr(i)$:
  $$\Pr(i) = \frac{w_i}{\sum_{i=1}^{m} w_i}$$

C. Simulation and results

We aim at computing how many messages must be sent to ensure that at least one message escapes from Eve. To this end, we implemented a simulator. We focus on the method, not the values that we compute. Besides, these values depend on the precise implementation of the routing algorithm, including the weights, that we don't include, nor discuss, here.

1) Simulation: We ran simulations varying the safety probability p_s and the distance d_{AB} between Alice and Bob. For each p_s, we generate a network with randomly spread eavesdroppers. For each distance d_{AB}, we generate 400 (Alice, Bob) pairs. For each such pair, we run 400 experiments. In each one we generate stochastic routes from Alice to Bob until we find a safe one (i.e., a route with no Eve). For each of the 400 experiments we gather the largest number of messages that were needed. Finally, we compute $N_0(p_s, d_{AB})$ (abbreviated N_0), the largest of these figures, i.e., the maximum number of messages that each 400×400 experiment required.

The critical value N_0 is such that:

$$\alpha(\pi_1, \ldots, \pi_{N_0}) = 1 \text{ for all of } 1.6 \times 10^5 \text{ trials}$$

Percolation theory tends to indicate that:

$$\alpha(\pi_1, \ldots, \pi_n) = \begin{cases} 1, & \text{if } n \geq N_0 \\ \tau : 0 \leq \tau < 1, & \text{if } n < N_0 \end{cases}$$

The routing algorithm may not be able to find any safe path in a reasonable amount of times, in particular when p_s is low, and d_{AB} is high. We set the maximum effort to $10,000$ times: Alice sends at most $10,000$ messages, if all are intercepted by Eve, then we declare that our routing algorithm is not capable of finding any safe path. Anyway, real world constraints (time and money) are likely to require a much smaller threshold. Section III explores solutions in such a case.

	p_s								
d	0.99	0.98	0.97	0.96	0.95	0.94	0.93	0.92	0.91
1	8	12	12	22	14	12	14	15	16
2	44	105	122	68	82	425	106	3058	×
3	87	51	273	99	122	233	439	1325	3472
4	95	171	160	408	244	1125	476	×	×
5	66	61	186	917	286	967	2149	1126	2803
6	34	397	356	377	644	583	921	3625	4806
7	43	194	155	395	625	420	2102	7038	3405
8	72	1645	224	414	936	773	1663	6011	7414
9	53	185	477	386	585	717	2794	×	4875
10	149	169	340	1267	3731	1267	2854	×	×
20	127	338	829	9300	×	×	×	×	×
30	315	1987	2908	×	×	×	×	×	×
40	386	4111	×	×	×	×	×	×	×
50	437	×	×	×	×	×	×	×	×
60	656	×	×	×	×	×	×	×	×
70	1911	×	×	×	×	×	×	×	×
80	3117	×	×	×	×	×	×	×	×
90	7039	×	×	×	×	×	×	×	×
100	4117	×	×	×	×	×	×	×	×
110	×	×	×	×	×	×	×	×	×
120	×	×	×	×	×	×	×	×	×

The symbol × stands for more than 10,000.

Fig. 6. Summary of experiment results for the worst cases

2) Results and analysis: Worst case study reveals $N_{max}(p_s, d_{AB})$ (abbreviated N_{max}) such that (at least for 1.6×10^5 trials):

$$\alpha(\pi_1, \ldots, \pi_n) = \begin{cases} 1, & \text{if } n \geq N_{max} \\ \tau : 0 \leq \tau < 1, & \text{if } n < N_{max} \end{cases}$$

Figure 6 gives the worst cases. Figure 7 plots this table and reveals a very chaotic behavior: this is because it presents the maximum number of messages needed, which is unbounded — percolation theory shows that the probability to require a large number drastically decreases, but not that it is bounded. The Figure 8 presents an averaged version of these values, and presents the expected regularity. Both pictures exhibit a critical value: $d_c(p_s)$ such that:

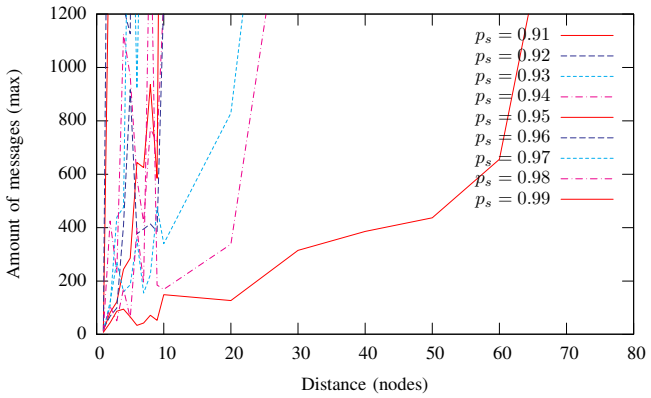
$$\alpha(\pi_1, \ldots, \pi_{N_{max}}) = \begin{cases} 1, & \text{if } d_{AB} \leq d_c(p_s) \\ \tau : 0 < \tau < 1, & \text{if } d_{AB} > d_c(p_s) \end{cases}$$

with:

- $d_c(p_s)$: critical value of distance with a given p_s.
- d_{AB}: distance between Alice and Bob.
- $N_{max}(p_s, d_{AB})$: the necessary amount of messages to be sent for the case of d_{AB} and p_s.

To look for this critical value $d_c(p_s)$, we look for the maximum distance between sender and receiver before the delivery is compromised (Figure 9). Figure 10 presents the percentage of safe messages arriving to destination, and Figure 11 focuses on the critical point from which the guarantee is lost.

In reality, the value of $d_c(p_s)$ depends on the budget reserved for the network construction. If one can afford 500 quantum messages, then almost-certain security is achieved up to $p_s < 0.97$, while 5000 messages tolerate p_s up to 0.95 (see



For each safety probability, given a distance between sender and receiver, this graph reports for 400×400 routings the maximal number of messages that had to be sent to deliver one safely. In other words, it exhibits the worst cases, hence its very chaotic behavior for low values of p_s. See Figure 8 for the average longest routings.

Fig. 7. The critical distance and the phase transition

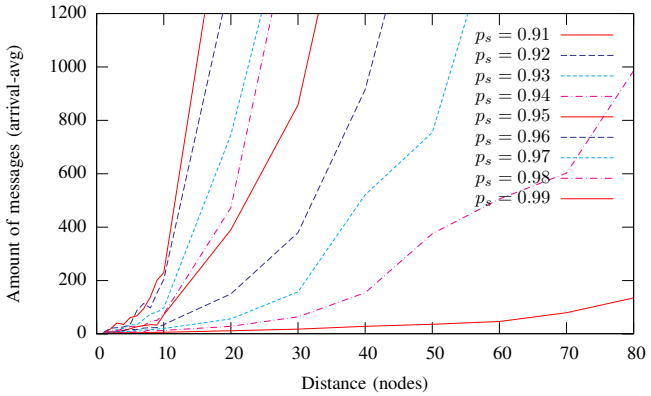


Figure 7 depicts the maximum number of messages needed to reach Alice from Bob safely, for 400 attempts and 400 (Alice, Bob) pairs. This graph averages over the 400 pairs of sender and receiver the maximum number of routings that were needed to find a safe one for 400 attempts. Values are ordered as expected, which helps to match lines with probabilities: $p = 0.99$ is near the bottom, $p = 0.98$ is immediately above, and so forth.

Fig. 8. Average longest paths

	p_s						
Attempts	0.99	0.98	0.97	0.96	0.95	0.93	0.91
500	50	20	10	9	5	4	1
10000	100	40	30	20	10	10	8

Maximum distance given a maximum number of attempts, and a probability safety.

Fig. 9. Critical distances

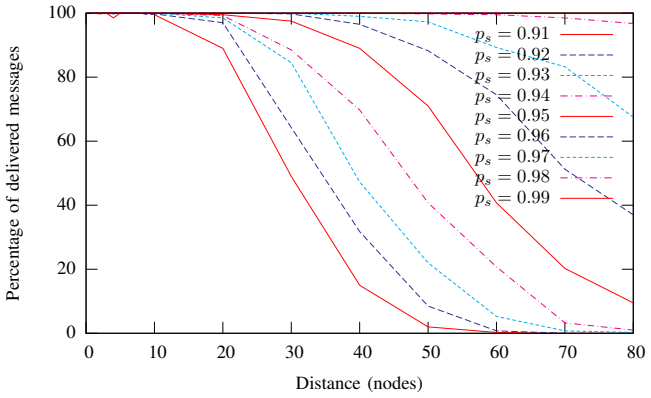
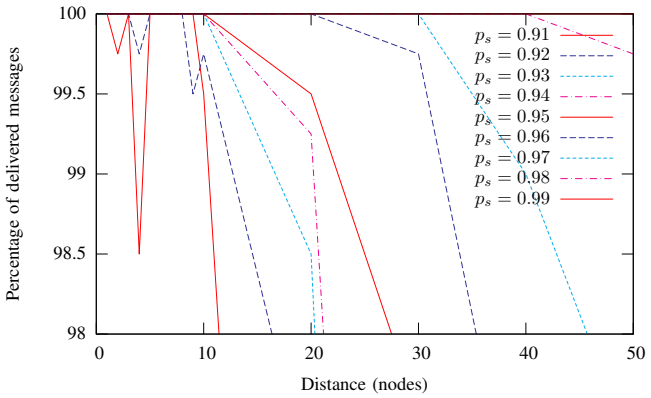


Fig. 10. Percentage of safe deliveries with 10,000 attempts



This graph focuses on the critical distances where safety is no longer ensured when making at most 10,000 attempts. The line for $p_s = 0.99$ is steadily equal to 100% until $d = 100$. This drawing exhibits the typical shape for percolation phenomena, see Figure 5.

Fig. 11. Safety for 10,000 attempts

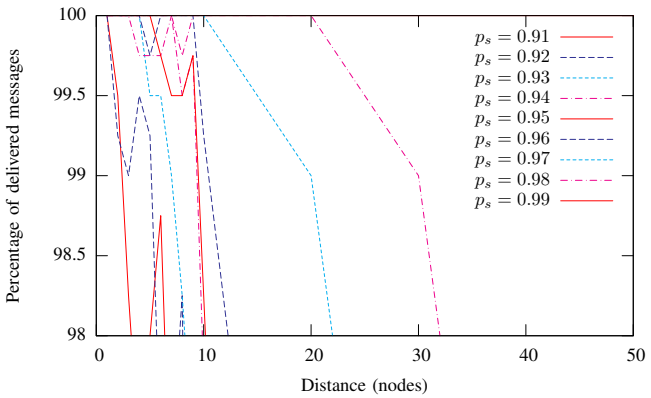


Fig. 12. Percentage of safe deliveries with 500 attempts

Figure 6). On the other hand, since all the messages are sent by QKD technology, the rate is low (about $R_0 = 1$Mbps). If one sends $N_0 = 500$ messages to obtain a secret key having the same length, then the transmission rate will be degraded, roughly by a factor of 500[1]:

$$R_r = \frac{R_0}{N_0} = \frac{1Mbps}{500} = 2Kbps$$

Similarly, with 5,000 the throughput is even further degraded. Thus, choosing an appropriate value for N_0 is not easy. It must be seriously analyzed when taking into account the balance between the budget and the performance of the targeted network. In this paper, we leave this objective as an open problem for future studies.

III. A GENERALIZED ROUTING ALGORITHM

A. The first remarks

The results obtained in Section II, about N_0 and d_c for each p_s, highlight that an improved routing algorithm is needed to cover large distances. For any pair, the first routing algorithm can successfully transmit a secret key provided that the distance does not exceed d_c. For distances greater than d_c, the necessary amount of messages sent by Alice sharply increases and becomes unrealistic. How can we alleviate this problem? Using a classical solution: relays. Intuitively, if the distance between two successive relays does not exceed d_c, then safe communication between them is ensured. Rather, security problems will arise from relays $R_1, R_2, \ldots$ on the path.

If we only consider relays on the path, then a path can be viewed as below:

$$Alice \rightarrow R_1 \rightarrow \cdots \rightarrow R_u \rightarrow Bob$$

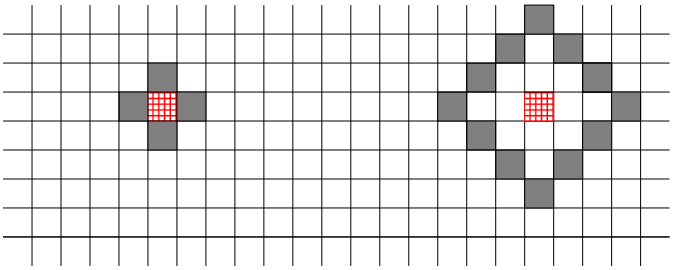
with:

1) $d(Alice, Bob) = d_{AB} > d_c$
2) $d(Alice, R_1) = d_c$
3) $d(R_i, R_{i+1}) = d_c$ for $i = 1, \ldots, u-1$
4) $d(R_u, Bob) \leq d_c$

Note that from the results of the previous sections, it is possible that:

- R_1 shares a common secret key $K_{A,1}$ with Alice, and a common secret key $K_{1,2}$ with R_2.
- $R_i, i = 2, \ldots, u-1$, shares a common secret key $K_{i-1,i}$ with R_{i-1}, and a common secret key $K_{i,i+1}$ with R_{i+1}.
- R_u shares a common secret key $K_{u-1,u}$ with R_{u-1}, and a common secret key $K_{u,B}$ with Bob.

[1]The precise figure depends on the contention nodes. Alice and Bob will limit the throughput, and since they have four neighbors, the penalty is somewhere between 125 and 500.



On the left-hand side, the neighborhood used in the first proposal (Section II), on the right-hand side, the one used in the relay searching problem.

Fig. 13. Neighborhood relations

Then, we can perform a classical transmission of the key K as the following:

$$\begin{aligned} Alice \ &\xrightarrow{K \oplus K_{A,1}} \ R_1 \\ &\xrightarrow{K \oplus K_{1,2}} \ R_2 \\ &\quad \ldots \\ &\xrightarrow{K \oplus K_{u,B}} \ Bob \end{aligned} \tag{3}$$

Bob can easily decipher the received message ($K \oplus K_{u,B}$) to the key K because he has the key $K_{u,B}$. If Eve controls only one relay of the chain, then she will also deduce the key K. But, if all the relays are safe, then Alice and Bob can totally trust the secrecy of the key K. Thus, the new problem arising is that we must have at least a chain of relays that contains only *safe* relays. This is similar to our first routing problem, except that the neighborhood relationship has been changed (see Figure 13). We can reuse the methodology developed in Section II: stochastic routing algorithm, simulator, and statistics to seek the number of searches that will return at least one chain of safe relays. However, the results can be reused directly. Although the neighborhood relationship is different, the probability that a node successfully chooses one safe node among its neighbors is not changed, it is still equal to p_s. The results of our first routing problem can be applied for the relays searching problem.

B. A generalized routing algorithm

We now consider our routing problem as a multi-layer routing problem. At Layer 0, the path is considered as a chain of nodes and the distance between two nodes is defined as the minimum intermediate nodes between these two nodes. Similarly, at Layer $i > 0$, the path is considered as a chain of relays and the distance is defined as the minimum necessary intermediate relays at this layer. The distance between two successive relays at Layer $i+1$ is always lower or equal to d_c if we view from Layer i, and lower or equal to $(d_c)^{(i+1)}$ nodes if we view from Layer 0. Clearly, the number of necessary layers will depend on the distance between Alice and Bob at Layer 0. More precisely, for a given p_s, we have:

- d_c: the critical value of distance for the cases of p_s (Figure 9);
- $d^{(0)}, \ldots, d^{(TL)}$: the distance between Alice and Bob at the layer $0, \ldots, TL$, defined as follows:
  1) $d^{(0)}$: number of intermediate nodes;
  2) $\forall i > 0$, we consider the number of relays at this layer as the unit of measurement, thus:
$$d^{(i)} = \lceil \frac{d^{(0)}}{\underbrace{d_c \times d_c \times \cdots \times d_c}_{i \text{ times}}} \rceil = \lceil \frac{d^{(0)}}{(d_c)^i} \rceil$$
  3) TL is such that:
$$TL = \max\{i : d^{(i)} > 1\}$$
  Note that $\forall i : d^{(i)} \leq d^{(i-1)} \times (d_c)$.
- $N^{(0)}, N^{(1)}, \ldots, N^{(TL)}$: the critical amount of finding a path between two successive relays at Layer $0, 1, \ldots, u$, respectively. Note that a path at Layer $i > 0$ is considered as a chain of relays of this layer. As the distance between two successive relays is always less than d_c, we have: $N^{(0)} = N^{(1)} = \cdots = N^{(TL)}$.

At the top layer we obtain a chain containing all the safe relays, or a safe path. With two successive relays, we consider the next layer $TL - 1$ and we can also obtain a safe path between them by using the same routing algorithm. Recursively we obtain a chain containing only safe nodes at layer 0. We can establish a perfect key between Alice and Bob. To simplify the presentation, we analyze an example of $d^{(0)} = (d_c)^2$ or $TL = 1$. We have two parameters:

- N_0: critical number of messages to send so that at least one message escapes from Eve if the distance between source and destination does not exceed d_c;
- N_1: the critical attempt of relay searches such that there is at least one chain that contains only safe relays.

The protocol of establishing the key between Alice and Bob is described below:

1) Alice sends N_1 messages to randomly establish N_1 chain of relays.
2) With each chain of relays $\{R_j^{(i)} : i = 1, \ldots, N_1;\ j = 1, \ldots, u_i\}$:
   - Establish, simultaneously, keys $K_1^{(i)}, \ldots, K_{u_i+1}^{(i)}$ between the $u_i + 1$ two-nodes pairs $(Alice; R_1), (R_1; R_2), \cdots, (R_{u-1}; R_u), (R_u; Bob)$, respectively. This task is done by the transmissions of N_0 random messages from $R_j^{(i)}$ to $R_{j+1}^{(i)}$ as described in the previous sections;
   - Alice and Bob establish a common key $K^{(i)}$ as described in Equation 3.
3) Alice and Bob compute the final secret key K:
$$K = \bigoplus_{i=1}^{N_1} K^{(i)}$$

Thus, each chain of relays must send N_0 random messages to establish a key with the first relay of this chain, and each relay on this chain must also send $N_0(p_s, d_c(p_s))$ random messages. The total amount of messages sent by Alice is:

$$\begin{aligned} N_{total} &= \underbrace{N_1 \times N_0}_{\text{to establish keys}} + \underbrace{N_1}_{\text{to establish relay chains}} \\ &= (N_0) \times (N_0 + 1) \end{aligned} \quad (4)$$

The above formula shows that the necessary amount of messages sent by Alice is a polynomial function of the distance (measured by nodes) between Alice and Bob. The exponential explosion at the critical distance d_c in our first routing algorithm (Figure 7) is avoided.

IV. Other work

A. Percolation theory

As mentioned in Section II-A, the context of percolation theory has many similarities with our problem context. The significant method used to solve percolation problems is simulations and statistics that report the percolation probability and an approximate formula that describes the system state at the phase transition. In this paper, our ambition is not to find approximate formulas that describe the evolution of a certain process. Some of the thresholds obtained by our simulator and statistics are enough to build our generalized secret key agreement scheme.

B. Stochastic routing

The main challenge in stochastic routings is how to compute the next-hop probabilities that optimize the routing cost. In some contexts, stochastic routing can be re-formalized as an abstract game between two players [6], [7]: the designer of the routing algorithm and the attacker that attempts to intercept packets. It is assumed that the attacker has a finite resource, i.e. she only intercepts packets at some nodes on the network. This is a zero-sum game in which a designer seeks a strategy to minimize the cost that he has to pay for a packet being safely transmitted and the attacker wants to maximize this cost. There are two types of games:

Off-line games
: The attacker starts by selecting where she will eavesdrop. This choice is made before the routing starts, but remains unknown to the designer. The designer must determine the next-hop probabilities that minimize the overall probability that the packet will be intercepted.

On-line games
: At every stage of game, two players are allowed to make their decisions simultaneously with full knowledge of current network state but without knowledge of what the other player will do.

Both have been solved in [7]. Though off-line routing games share common assumptions with our problem, the final goal is different: In off-line routing games, the main task is to determine the next-hop probabilities that minimize the overall probability of packet interception, while in our

problem, the goal is to succeed in getting at least one packet safely transmitted. Previous works on stochastic routing focus on performance metrics (latency, throughput, acceptance rate, etc.), which are not of major importance to quantum networks. What matters is sensitivity to eavesdropping and security.

C. Quantum network

The first quantum network, DARPA Quantum Network, was built to test the strength of such systems in the real-world applications. It consists of three sites (Cambridge, Harvard, and Boston University) and became fully operational in October 2003 [10]. It relies on trusted relays: one must trust the security of all the participants, and be sure that eavesdroppers cannot sniff any information on any of the nodes. In realistic contexts, nobody can be sure that he does not reveal any information for eavesdroppers. Moreover, in a larger quantum network, the assumption that one can trust all the nodes becomes unacceptable. In this paper, we studied large quantum networks in a more realistic context: nodes are not totally trustworthy, there is a probability that nodes are controlled by eavesdroppers.

V. Conclusion

We investigated the constraints of quantum networks and the ineluctable probability that some nodes are attacked. We proposed a secure key exchange scheme that scales well with distance. It is based on stochastic routing, and was analyzed using percolation-theory based methods. Not only did it validate our solution, it also gave figures allowing us to engineer various parameters. For instance, given the probability that nodes are attacked and the distance between source and destination, it gives the the number of pieces the message must be broken into.

Much remains to be done. Studying more general topologies is of primary importance: grids are only the first stab. The node safety-probability might also vary between regions. Finding formulas (explicit or implicit via equations) is also of interest, as they usually provide more revealing results than simulations do. Finally, we will also work to improve our stochastic routing proposal.

Acknowledgment

Stephen Frank proofread this paper.

References

[1] C. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in *Proc. of IEEE Int. Conf. on Computers, Systems, and Signal*, Bangalore, India, 1984.

[2] P. Shor and J. Preskill, "Simple proof of security of the BB84 quantum key distribution protocol," *Phys. Rev. Lett.*, vol. 85, 2000.

[3] C. Elliott, D. Pearson, and G. Troxel, "Quantum cryptography in practice," in *Proc. of the Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Karlsruhe, Germany, 2003.

[4] T. Kimura, Y. Nambu, T. Hatanaka, A. Tomita, H. Kosaka, and K. Nakamura, "Single-photon interference over 150-km transmission using silica-based integrated-optic interferometers for quantum cryptography criterion," *Jap. J. of Appl. Phy.*, vol. 43, 2004.

[5] C. Shannon, "Communication theory of secrecy of systems," *Bell System Technical Journal*, 1949.

[6] S. Bohacek, J. P. Hespanha, and K. Obraczka, "Saddle policies for secure routing in communication networks," in *Proc. of the 41st IEEE Conf. on Decision and Control*, 2002.

[7] S. Bohacek, J. P. Hespanha, K. Obraczka, J. Lee, and C. Lim, "Enhancing security via stochastic routing," in *Proc. 11th Int. Conf. on Computer Communication and Networks*, 2002.

[8] V. Beffara and V. Sidoravius, "Percolation theory," *Encyclopedia of Mathematical Physics*, 2006.

[9] S. Amor, D. H. Tran, and M. Bui, "A percolation based model for ATC simulation," in *Proc. of the 4th Int. Conf. on Computer Sciences, Research Innovation and Vision for the Futur*, Vietnam, 2006.

[10] C. Elliott, A. Colvin, D. Pearson, O. Pikalo, J. Schlafer, and H. Yeh, "Current status of the DARPA quantum network," 2005. [Online]. Available: http://arxiv.org/abs/quant-ph/0503058